Using JavaScript Monitoring to Prevent Device Fingerprinting

by Nataliia Bielova, Frédéric Besson and Thomas Jensen, Inria

Today's Web users are continuously tracked as they browse the Web. One of the techniques for tracking is device fingerprinting that distinguishes users based on their Web browser and operating system properties. We propose solutions to detect and prevent device fingerprinting via runtime monitoring of JavaScript programs.

The use of sophisticated web tracking technologies has grown enormously in the last decade. Advertisement companies and tracking agencies are collecting increasing amounts of data about Web users in order to better advertise their products. Social media plugins also collect data to learn about online habits and preferences of their users. In the last five years, researchers have started to examine the mechanisms used for Web tracking. Recent research has shown that advertising agencies and networks use a wide range of techniques in order to track users across the Web.

Web tracking via cookies is well known. Cookies are stored in a user's browser so that the tracking script can immediately recognise the user. However, another group of tracking techniques, called 'device fingerprinting', does not require storing anything in a user's browser. Fingerprinting scripts make a snapshot of the configuration of the Web browser and operating system properties and then are able to distinguish a particular user from all other website visitors. Unlike cookies, this technique also works perfectly across sites, meaning that the tracker will know all the web sites that the user has visited if this tracker's script is present on these sites. The Panopticlick project by Electronic Frontier Foundation was the first to demonstrate the power of fingerprinting in 2010. Since then, researchers have found new ways to distinguish Web users, for example through HTML5 Canvas fingerprinting, which was discovered only in 2012.

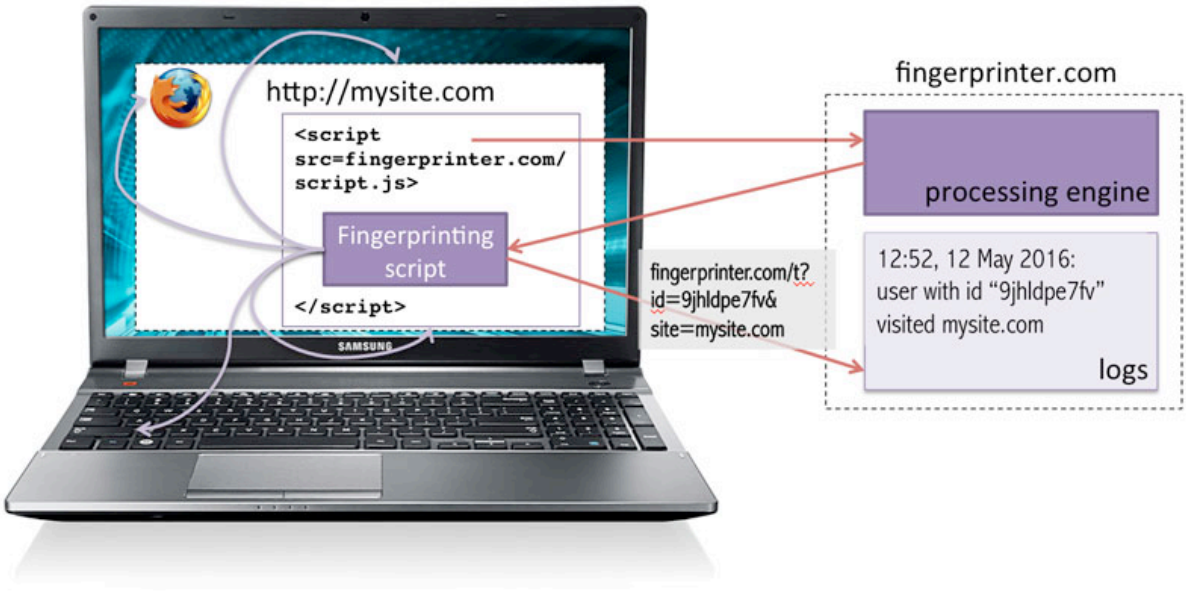

Figure 1: Device fingerprinting: a fingerprinting script collects data about Web browser and operating system properties, such as Web browser version, list of installed plugins, screen resolution, time zone etc., encodes it into a string and sends it back to fingerprinter.com.

Within the French ANR projects SecCloud (Security of cloud programming) and AJACS (Analyses of JavaScript Applications: Certification and Security), in INRIA, we have proposed a new solution to protect Web users from being fingerprinted. We are developing a tool that

formally guarantees that the scripts run in a browser are not fingerprinting the user. This can be done either by detecting and blocking tracking scripts, or by modifying their tracking behaviour. To do so, we developed a monitor that analyses a potentially tracking script, and computes how much fingerprinting information this script collects. The more information it collects, the more easily it can distinguish the user from all other visitors.

As a first step, we have developed a methodology to analyse how much identifiable information a tracker may learn about a user through an execution of a script. While a script runs, it collects some data about the Web browser and operating system configuration and sends this data back to the server. How much identifiable information did the tracker learn by observing this data? We have shown that this problem can be stated as an information-flow problem that answers the question: what is the probability that a server can identify a user after analysing the output of the script? If the probability is low, the user is unlikely to be tracked. If the probability is high, this is a tracking script trying to identify the user.

We have also developed a quantitative information flow monitor [1] computing how much the tracker learns when running a script in the user's browser. The monitor uses a combination of dynamic and static analysis and over-approximates on-the-fly the amount of information that is learnt by running the script. If the amount of information is below a threshold, it is safe to send the output to the server. Otherwise, counter-measures need to be taken, such as shutting down the connection or providing forged but credible output. The theoretical foundations of such browser randomisation were developed in [2].

Next, we recently proposed a new version of a quantitative information flow monitor that is more precise in computing the knowledge of the tracker [3]. This new version expresses the monitoring of attacker knowledge in a general framework of semantics-based program analysis, and shows how a knowledge monitor can be combined with existing monitoring techniques for information flow control, such as the 'no-sensitive-upgrade' principle.

Link:
https://panopticlick.eff.org/

References:
[1] F. Besson, N. Bielova, T. Jensen: "Hybrid Information Flow Monitoring Against Web Tracking", IEEE CSF 2013.
[2] F. Besson, N. Bielova, T. Jensen: "Browser Randomisation against Fingerprinting: A Quantitative Information Flow Approach", NordSec 2014.
[3] F. Besson, N. Bielova, T. Jensen: "Hybrid Monitoring of Attacker Knowledge", IEEE CSF 2016.

Please contact:
Nataliia Bielova, Inria, France
+33 4 92 38 77 87
nataliia.bielova@inria.fr